

FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL IIIII
LLLLLLLLLLL IIIII
SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```



```
0001 0 MODULE SNDMSB (
0002 0     LANGUAGE (BLISS),
0003 0     IDENT = 'V04-000',
0004 0     ) =
0005 1 BEGIN
0006 1
0007 1
0008 1 *****
0009 1 *
0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 * ALL RIGHTS RESERVED.
0013 1 *
0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 * TRANSFERRED.
0020 1 *
0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 * CORPORATION.
0024 1 *
0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *****
0029 1
0030 1
0031 1 ++
0032 1
0033 1 FACILITY: F11ACP Structure Level 1
0034 1
0035 1 ABSTRACT:
0036 1
0037 1     This routine constructs and sends a message to the symbiont manager
0038 1     to cause a file to be spooled and deleted.
0039 1
0040 1 ENVIRONMENT:
0041 1
0042 1     STARLET operating system, including privileged system services
0043 1     and internal exec routines.
0044 1
0045 1 --
0046 1
0047 1
0048 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 5-Jun-1978 11:23
0049 1
0050 1 MODIFIED BY:
0051 1
0052 1     V03-006 ACG0346      Andrew C. Goldstein,      1-Aug-1983 15:26
0053 1     Convert back to use in F11A ACP
0054 1
0055 1     V03-005 CDS0004      Christian D. Saether      26-Jul-1983
0056 1     Use new send job controller service.
0057 1
```



```

58 0058 1 V03-004 CDS0003 Christian D. Saether 13-May-1983
59 0059 1 Reflect change to IOC$CVT_DEVNAM interface.
60 0060 1
61 0061 1 V03-003 CWH1002 CW Hobbs 1-Mar-1983
62 0062 1 Use extended pid and owner in symbiont message
63 0063 1
64 0064 1 V03-002 CDS0002 Christian D. Saether 16-Dec-1982
65 0065 1 Make item list generation pic.
66 0066 1
67 0067 1 V03-001 CDS0001 C Saether 30-Jul-1982
68 0068 1 Changes for ACP to XQP.
69 0069 1 No timer on waiting for job controller reply.
70 0070 1
71 0071 1 V02-004 ACG0245 Andrew C. Goldstein, 23-Dec-1981 21:21
72 0072 1 Check error return from queue manager
73 0073 1
74 0074 1 V02-003 SPF0025 Steve Forgey 08-Sep-1981
75 0075 1 Add new header fields to symbiont manager message.
76 0076 1
77 0077 1 V02-001 GWF0043 Gary W. Fowler 12-May-1981 15:20
78 0078 1 Add file size option and file size to message sent to job
79 0079 1 controller.
80 0080 1
81 0081 1 V02-000 ACG0167 Andrew C. Goldstein, 16-Apr-1980 19:25
82 0082 1 Previous revision history moved to [F11B.SRC]F11B.REV
83 0083 1 **
84 0084 1
85 0085 1
86 0086 1 LIBRARY 'SYSS$LIBRARY:LIB.L32';
87 0087 1 REQUIRE 'SRCS:FCPDEF.B32';
88 0402 1
89 0403 1
90 0404 1 MACRO
91 0405 1
92 0406 1 ! Layout of item list for job controller.
93 0407 1 ! The first item is the queue name.
94 0408 1 ! The second item is the file identification.
95 0409 1 ! The third item is the delete file item code.
96 0410 1 ! Only fields that are filled in with non-zero values are defined.
97 0411 1
98 0412 1
99 0413 1 ITM_QNAMSIZ = 0, 0, 16, 0 %, ! queue name size
100 0414 1 ITM_QNAMCODE = 2, 0, 16, 0 %, ! queue name item code
101 0415 1 ITM_QNAMADDR = 4, 0, 32, 0 %, ! queue name address
102 0416 1
103 0417 1 ITM_FILEINFOSIZ = 12, 0, 16, 0 %, ! file identification size
104 0418 1 ITM_FILEINFOCODE = 14, 0, 16, 0 %, ! file id item code
105 0419 1 ITM_FILEINFOADDR = 16, 0, 32, 0 %, ! file id address
106 0420 1
107 0421 1 ITM_DELFILECODE = 26, 0, 16, 0 %, ! delete file item code.
108 0422 1
109 0423 1 ITM_USERIDSIZ = 36, 0, 16, 0 %, ! user ID block size
110 0424 1 ITM_USERIDCODE = 38, 0, 16, 0 %, ! user ID block item code
111 0425 1 ITM_USERIDADDR = 40, 0, 32, 0 %, ! user ID block address
112 0426 1
113 0427 1
114 0428 1 LITERAL

```



```

115      0429 1      ITM_LENGTH      = 4*12 + 4      ! 4 item codes + stopper.
116      0430 1      FS_LENGTH       = 16 + 2*FIDISC_LENGTH + 4 + 4 + 2
117      0431 1      + 16 + 2 + 20,
118      0432 1      UID_LENGTH       = 4 + 8 + 1 + 12;      ! UIC + account + priority + username

```



```

120 0433 1 GLOBAL ROUTINE SEND_SYMBIONT (HEADER, FCB) : NOVALUE =
121 0434 1
122 0435 1 ++
123 0436 1
124 0437 1 FUNCTIONAL DESCRIPTION:
125 0438 1
126 0439 1 This routine constructs and sends a message to the symbiont manager
127 0440 1 to cause a file to be spooled and deleted.
128 0441 1
129 0442 1
130 0443 1 CALLING SEQUENCE:
131 0444 1 SEND_SYMBIONT (ARG1, ARG2)
132 0445 1
133 0446 1 INPUT PARAMETERS:
134 0447 1 ARG1: address of file header
135 0448 1 ARG2: address of file control block
136 0449 1
137 0450 1 IMPLICIT INPUTS:
138 0451 1 IO_PACKET: address of I/O packet of this request
139 0452 1
140 0453 1 OUTPUT PARAMETERS:
141 0454 1 NONE
142 0455 1
143 0456 1 IMPLICIT OUTPUTS:
144 0457 1 NONE
145 0458 1
146 0459 1 ROUTINE VALUE:
147 0460 1 NONE
148 0461 1
149 0462 1 SIDE EFFECTS:
150 0463 1 message sent to symbiont manager
151 0464 1
152 0465 1 --
153 0466 1
154 0467 2 BEGIN
155 0468 2
156 0469 2 MAP
157 0470 2 HEADER : REF BBLOCK, ! file header arg
158 0471 2 FCB : REF BBLOCK; ! file control block arg
159 0472 2
160 0473 2 BUILTIN
161 0474 2 LOCC;
162 0475 2
163 0476 2 LINKAGE
164 0477 2 L_IOC_CVT = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 5,
165 0478 2 REGISTER = 4; REGISTER = 1) :
166 0479 2 NOTUSED (6, 7, 8, 9, 10, 11);
167 0480 2
168 0481 2 LOCAL
169 0482 2 ITEMLIST : BBLOCK [ITM_LENGTH], ! item list for jbc.
170 0483 2 FILEINFO_STR : VECTOR [FS_LENGTH, BYTE], ! everything the jbc wants
171 0484 2 ! to know about the file.
172 0485 2 USERID_BLOCK : VECTOR [UID_LENGTH, BYTE], ! user identification block
173 0486 2 P, ! string scan pointer
174 0487 2 JBCSTS : VECTOR [2], ! io status block for SNDJBC
175 0488 2 LENGTH, ! length of converted device name
176 0489 2 IDENT_AREA : REF BBLOCK, ! address of file header ident area

```



```

177 0490 2          PCB          : REF BBLOCK,      ! user's PCB
178 0491 2          JIB          : REF BBLOCK,      ! user's JIB
179 0492 2          UCB          : REF BBLOCK,      ! UCB of spooled device
180 0493 2          VCB          : REF BBLOCK;      ! VCB of spooled device
181 0494 2
182 0495 2  EXTERNAL
183 0496 2          CLEANUP_FLAGS : BITVECTOR,      ! cleanup action flags
184 0497 2          USER_STATUS  : VECTOR,         ! status returned to user
185 0498 2          IO_PACKET    : REF BBLOCK,      ! I/O packet of user request
186 0499 2          CURRENT_UCB  : REF BBLOCK,      ! UCB of current device
187 0500 2          SCH$GL_PCBVEC : REF VECTOR ADDRESSING_MODE (GENERAL);
188 0501 2                                     ! system PCB vector
189 0502 2
190 0503 2  EXTERNAL ROUTINE
191 0504 2          MAKE_STRING,      ! convert RAD-50 to name string
192 0505 2          IOC$CVT_DEVNAM : L_IOC_CVT ADDRESSING_MODE (GENERAL);
193 0506 2                                     ! get device name of UCB
194 0507 2
195 0508 2  ! Initialize item list to zeroes.
196 0509 2  !
197 0510 2
198 0511 2  CH$FILL (0, ITM_LENGTH, ITEMLIST);
199 0512 2
200 0513 2  ! Get UCB and VCB addresses for the spooled device.
201 0514 2  !
202 0515 2
203 0516 2  UCB = .IO_PACKET[IRP$$_MEDIA];
204 0517 2  VCB = .UCB[UCB$$_VCB];
205 0518 2
206 0519 2  ! Point the first item at the queue name in the spooled device's VCB.
207 0520 2  ! This might be a little racy in that we could become unscheduled and
208 0521 2  ! the device set non-spoiled or changed before the sndjbc service gets
209 0522 2  ! a chance to look at it, but that should be no more harmful than if
210 0523 2  ! we copied it off to local storage and it changed before the service
211 0524 2  ! executed anyway. At worst the VCB gets deallocated and some garbage
212 0525 2  ! is picked up for the name. BFD.
213 0526 2  !
214 0527 2
215 0528 2  ITEMLIST [ITM_QNAMSIZ] = .VCB [VCB$$_QNAMECNT];
216 0529 2  ITEMLIST [ITM_QNAMCODE] = SJCS_QUEUE;
217 0530 2  ITEMLIST [ITM_QNAMADDR] = VCB [VCB$$_QNAMECNT] + 1;
218 0531 2
219 0532 2  ! Fill in the file identification string.
220 0533 2  ! "Normal" callers of this service supply only the counted device
221 0534 2  ! string, the file ID and the directory ID. The service then performs
222 0535 2  ! an access function on that file to recover other information it needs
223 0536 2  ! in a trustworthy manner.
224 0537 2  ! However, we ARE the file system, and therefore can be trusted. Also,
225 0538 2  ! an attempt to call us back recursively with an access function just
226 0539 2  ! flat out will not work, because we'll be sitting here waiting for the
227 0540 2  ! SNDJBC service to finish.
228 0541 2  ! Therefore, we will also put the end of file block, access mask,
229 0542 2  ! and expanded file spec in this string also. SNDJBC will look at it
230 0543 2  ! because we are calling from exec mode or better and use this info
231 0544 2  ! rather than doing an access function on the file.
232 0545 2
233 0546 2

```



```

: 234 0547 2 ! First get the counted device string. Leave the trailing ":" on
: 235 0548 2 ! the string for now, because we'll want it a little later when
: 236 0549 2 ! the full file name string is built.
: 237 0550 2 !
: 238 0551 2 !
: 239 0552 2 IOC$CVT_DEVNAM (15, FILEINFO_STR [1],
: 240 0553 2 .CURRENT_UCB, 0; LENGTH);
: 241 0554 2 FILEINFO_STR [0] = .LENGTH;
: 242 0555 2 !
: 243 0556 2 ! Pick up the file ID from the FCB instead of the file header
: 244 0557 2 ! because the RVN has already been normalized in the FCB, i.e.,
: 245 0558 2 ! APPLY_RVN has already been called.
: 246 0559 2 ! Also zero out the DID while we're at it.
: 247 0560 2 !
: 248 0561 2 !
: 249 0562 2 CH$COPY (FID$C_LENGTH, FCB[FCB$W_FID], 0, 2*FID$C_LENGTH, FILEINFO_STR [16]);
: 250 0563 2 !
: 251 0564 2 ! Next is the end of file block for the file.
: 252 0565 2 !
: 253 0566 2 !
: 254 0567 2 (FILEINFO_STR [28])<0,32> = ROT (.BBLOCK[HEADER[FH1$W_RECATTR],FAT$L_EFBLK], 16);
: 255 0568 2 IF .(FILEINFO_STR [28])<0,32> NEQ 0
: 256 0569 2 AND .BBLOCK[HEADER[FH1$W_RECATTR],FAT$W_FFBYTE] EQL 0
: 257 0570 2 THEN
: 258 0571 2 (FILEINFO_STR [28])<0,32> = .(FILEINFO_STR [28])<0,32> - 1;
: 259 0572 2 !
: 260 0573 2 ! File access mask (everything allowed).
: 261 0574 2 !
: 262 0575 2 !
: 263 0576 2 (FILEINFO_STR [32])<0,32> = %X'FFFFFFFF';
: 264 0577 2 !
: 265 0578 2 ! Build an expanded file name string including the device, null directory
: 266 0579 2 ! spec, and the file name.
: 267 0580 2 !
: 268 0581 2 !
: 269 0582 2 P = CH$MOVE (.FILEINFO_STR [0], FILEINFO_STR [1], FILEINFO_STR [38]);
: 270 0583 2 !
: 271 0584 2 ! Now that the entire device spec has been copied, adjust the count
: 272 0585 2 ! on the first string to drop the trailing ":", just like RMS does
: 273 0586 2 ! for the DVI field in the NAM block.
: 274 0587 2 !
: 275 0588 2 !
: 276 0589 2 FILEINFO_STR [0] = .FILEINFO_STR [0] - 1;
: 277 0590 2 !
: 278 0591 2 ! Add in null directory spec.
: 279 0592 2 !
: 280 0593 2 !
: 281 0594 2 (.P)<0,16> = '[';
: 282 0595 2 P = .P + 2;
: 283 0596 2 !
: 284 0597 2 ! Now pick up the file name from the ident area in the header.
: 285 0598 2 !
: 286 0599 2 !
: 287 0600 2 IDENT_AREA = .HEADER + .HEADER[FH1$B_IDOFFSET] * 2;
: 288 0601 2 P = .P + MAKE_STRING (IDENT_AREA[F11$W_FILENAME] - 6, .P);
: 289 0602 2 !
: 290 0603 2 (FILEINFO_STR [36])<0,16> = .P - FILEINFO_STR [38];

```



```

: 291      0604      2
: 292      0605      2 ! Set up the item list entry for the file information string.
: 293      0606      2
: 294      0607      2
: 295      0608      2 ITEMLIST [ITM_FILEINFOSIZ] = .P - FILEINFO_STR [0];
: 296      0609      2 ITEMLIST [ITM_FILEINFOCODE] = SJC$ FILE_IDENTIFICATION;
: 297      0610      2 ITEMLIST [ITM_FILEINFOADDR] = FILEINFO_STR [0];
: 298      0611      2
: 299      0612      2 ! Build the user identification block, containing UIC, account, base
: 300      0613      2 ! priority, and username.
: 301      0614      2
: 302      0615      2
: 303      0616      2 PCB = .SCH$GL PCBVEC[.(IO_PACKET[IRP$L_PID])<0,16>];
: 304      0617      2 JIB = .PCB[PCB$L_JIB];
: 305      0618      2 (USERID_BLOCK[00])<0,32> = .PCB[PCB$L_UIC];
: 306      0619      2 (USERID_BLOCK[24])<0,08> = .PCB[PCB$L_PRI];
: 307      0620      2 $ASSUME (JIB$L_USERNAME, EQL, 12);
: 308      0621      2 CH$MOVE (JIB$L_USERNAME, JIB[JIB$L_USERNAME], USERID_BLOCK[04]);
: 309      0622      2 $ASSUME (JIB$L_ACCOUNT, EQL, 8);
: 310      0623      2 CH$MOVE (JIB$L_ACCOUNT, JIB[JIB$L_ACCOUNT], USERID_BLOCK[16]);
: 311      0624      2
: 312      0625      2 ! Build the item descriptor for the user identification.
: 313      0626      2
: 314      0627      2
: 315      0628      2 ITEMLIST [ITM_USERIDSIZ] = 25;
: 316      0629      2 ITEMLIST [ITM_USERIDCODE] = SJC$ USER_IDENTIFICATION;
: 317      0630      2 ITEMLIST [ITM_USERIDADDR] = USERID_BLOCK;
: 318      0631      2
: 319      0632      2 ! Finally the item code to delete the file after printing.
: 320      0633      2
: 321      0634      2
: 322      0635      2 ITEMLIST [ITM_DELFILECODE] = SJC$_DELETE_FILE;
: 323      0636      2
: 324      0637      2 ! The status from the service is always written to the iosb and the
: 325      0638      2 ! completion ast is always delivered, regardless of status, so we
: 326      0639      2 ! do not need to separately check the status of the service call.
: 327      0640      2
: 328      0641      2
: 329      0642      2 $SNDJBCW (EFN = EFN,
P      0643      2     FUNC = SJC$_ENTER_FILE,
P      0644      2     IOSB = JBCSTS,
P      0645      2     ITMLST = ITEMLIST);
: 333      0646      2
: 334      0647      2 ! A full longword of status is returned from sndjbc.
: 335      0648      2
: 336      0649      2
: 337      0650      2 IF NOT .JBCSTS [0]
: 338      0651      2 THEN
: 339      0652      2     BEGIN
: 340      0653      2     CLEANUP_FLAGS[CLF_DELFILE] = 1;
: 341      0654      2     USER_STATUS[1] = .JBCSTS [0];
: 342      0655      2     ERR_EXIT (SS$_NOTPRINTED);
: 343      0656      2     END;
: 344      0657      2
: 345      0658      1 END;

```

! end of routine SEND_SYMBIONT

; R


```

34      00      5E      FF5C      CE      007C 00000
6E      70      AE      00      2C 00007
          38      AO      DO 0000E
          34      AO      DO 00013
70      AE      0B      AO      9B 0001B
72      AE      86      8F      9B 00020
74      AE      0C      AO      9E 00025
          51      25      AE      9E 0002A
          54      D4      0002E
          55      0000G      CF      DO 00030
          50      0F      DO 00035
          00000000G      00      16 00038
24      AE      51      90 0003E
          50      08      AC      DO 00042
OC      00      24      AO      06      2C 00046
          34      AE      0004C
          50      04      AC      DO 0004E
40      AE      16      AO      10      9C 00052
          08      13 00058
          1A      AO      B5 0005A
          03      12 0005D
          40      AE      D7 0005F
          44      AE      01      CE 00062 1$:
          50      24      AE      9A 00066
4A      AE      25      AE      50      28 0006A
          24      AE      97 00070
          83      5D5B      8F      B0 00073
          50      04      BC      9A 00078
          50      04      BC40      3E 0007C
          53      DD 00081
          FA      AO      9F 00083
          0000G      CF      02      FB 00086
          53      50      C0 0008B
          50      4A      AE      9E 0008E
48      AE      53      50      A3 00092
          50      24      AE      9E 00097
          53      50      A3 0009B
          7E      AE      27      B0 000A0
          DC      AD      24      AE      9E 000A4
          51      00000000G      00      DO 000A9
          50      0000G      CF      DO 000B0
          50      0C      C0 000B5
          50      60      3C 000B8
          50      6140      DO 000BB

```

```

.TITLE SNDSMB
.IDENT \V04-000\

.EXTRN CLEANUP_FLAGS, USER_STATUS
.EXTRN IO_PACKET, CURRENT_OCB
.EXTRN SCH$GL_PCBVEC, MAKE_STRING
.EXTRN IOC$CVT_DEVNAM, SYS$SNDJBCW

.PSECT $CODE$,NOWRT,2

.ENTRY SEND_SYMBIONT, Save R2,R3,R4,R5,R6      : 0433
MOVAB -164(SP), SP
MOVCS #0, (SP), #0, #52, ITEMLIST      : 0511

MOVL IO_PACKET, R0      : 0516
MOVL 56(R0), UCB
MOVL 52(UCB), VCB      : 0517
MOVZBW 11(VCB), ITEMLIST      : 0528
MOVZBW #134, ITEMLIST+2      : 0529
MOVAB 12(R0), ITEMLIST+4      : 0530
MOVAB FILEINFO_STR+1, R1      : 0552
CLRL R4
MOVL CURRENT_UCB, R5
MOVL #15, R0
JSB IOC$CVT_DEVNAM
MOVB LENGTH, FILEINFO_STR      : 0554
MOVL FCB, R0      : 0562
MOVCS #6, 36(R0), #0, #12, FILEINFO_STR+16

MOVL HEADER, R0      : 0567
ROTL #16, 22(R0), FILEINFO_STR+28
BEQL 1$      : 0568
TSTW 26(R0)      : 0569
BNEQ 1$
DECL FILEINFO_STR+28      : 0571
MNEGL #1, FILEINFO_STR+32      : 0576
MOVZBL FILEINFO_STR, R0      : 0582
MOVCS R0, FILEINFO_STR+1, FILEINFO_STR+38
DECB FILEINFO_STR      : 0589
MOVW #23899, (P)+      : 0594
MOVZBL @HEADER, R0      : 0600
MOVAV @HEADER(R0), IDENT_AREA
PUSHL P      : 0601
PUSHAB -6(IDENT_AREA)
CALLS #2, MAKE_STRING
ADDL2 R0, P      : 0603
MOVAB FILEINFO_STR+38, R0
SUBW3 R0, P, FILEINFO_STR+36      : 0608
MOVAB FILEINFO_STR, R0
SUBW3 R0, P, ITEMLIST+12
MOVW #39, ITEMLIST+14      : 0609
MOVAB FILEINFO_STR, ITEMLIST+16      : 0610
MOVL SCH$GL_PCBVEC, R1      : 0616
MOVL IO_PACKET, R0
ADDL2 #12, R0
MOVZWL (R0), R0
MOVL (R1)(R0), PCB

```


		56	0080	C0	D0	000BF	MOVL	128(PCB), JIB	: 0617
		08	00BC	C0	D0	000C4	MOVL	188(PCB), USERID_BLOCK	: 0618
		20	2F	A0	90	000CA	MOVB	47(PCB), USERID_BLOCK+24	: 0619
OC	AE	0C	A6	0C	28	000CF	MOVC3	#12, 12(JIB), USERID_BLOCK+4	: 0621
18	AE	18	A6	08	28	000D5	MOVC3	#8, 24(JIB), USERID_BLOCK+16	: 0623
		F0	AD	8F	D0	000DB	MOVL	#9830425, ITEMLIST+36	: 0628
		F4	AD	AE	9E	000E3	MOVAB	USERID_BLOCK, ITEMLIST+40	: 0630
		E6	AD	18	B0	000E8	MOVW	#24, ITEMLIST+26	: 0635
				7E	7C	000EC	CLRQ	-(SP)	: 0645
			08	AE	9F	000EE	PUSHAB	JBCSTS	
			7C	AE	9F	000F1	PUSHAB	ITEMLIST	
		7E		13	7D	000F4	MOVQ	#19, -(SP)	
				01	DD	000F7	PUSHL	#1	
00000000G		00		07	FB	000F9	CALLS	#7, SYSSNDJBCW	
		0E		6E	E8	00100	BLBS	JBCSTS, 2\$	: 0650
0000G		CF		20	88	00103	BISB2	#32, CLEANUP_FLAGS+2	: 0653
0000G		CF		6E	D0	00108	MOVL	JBCSTS, USER_STATUS+4	: 0654
			2184	8F	BF	0010D	CHMU	#8580	: 0655
				04	00111	2\$:	RET		: 0658

; Routine Size: 274 bytes, Routine Base: \$CODE\$ + 0000

: 346 0659 1
: 347 0660 1 END
: 348 0661 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	274	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	29	0	1000	00:01.9

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:SNDSMB/OBJ=OBJ\$:SNDSMB MSRC\$:SNDSMB/UPDATE=(ENH\$:SNDSMB)

SNDSMB
V04-000

K 2
16-Sep-1984 01:19:31
14-Sep-1984 12:29:53

VAX-11 Bliss-32 V4.0-742
DISK\$VMMASTER:[F11A.SRC]SNDSMB.B32;1 Page 10
(2)

; Size: 274 code + 0 data bytes
; Run Time: 00:10.8
; Elapsed Time: 00:32.3
; Lines/CPU Min: 3682
; Lexemes/CPU-Min: 19883
; Memory used: 133 pages
; Compilation Complete

TRU
V04

: R

...

...

...

0167 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

